МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
  
 к курсовой работе по дисциплине «Программная инженерия»

по теме «Клавиатурный тренажёр с функциями администратора»

Обучающийся С.А. Атякшев

Обучающийся Д.О. Колбанов

Обучающийся Р.К. Лукашевич

Руководитель Л.С. Зеленко

Самара 2024

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

ЗАДАНИЕ

на курсовую работу по дисциплине

«Программная инженерия»

обучающимся в группе № 6401-020302D

С.А. Атякшеву

Д.О. Колбанову

Р.К. Лукашевичу

1. Тема проекта:«Клавиатурный тренажёр с функциями администратора»
2. Исходные данные к проекту**:** см. приложение к заданию
3. Перечень вопросов, подлежащих разработке:
   1. Произвести анализ предметной области: изучить основные принципы составления упражнений для клавиатурного тренажёра
   2. Выполнить обзор существующих систем-аналогов
   3. Разработать информационно-логический проект системы по методологии UML
   4. Разработать и реализовать программное и информационное обеспечение, провести его тестирование и отладку
   5. Оформить документацию курсовой работы
   6. Подготовить презентацию по разработанной системе
4. Перечень графических разработок:
   1. Структурная схема системы
   2. Канонические диаграммы UML
   3. Схемы основных алгоритмов
5. Календарный план выполнения работ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Содержание работы по этапам | Объем этапа в % к общему объему проекта | Срок  окончания | Фактическое выполнение |
| 1 | Оформление технического задания и его утверждение | 5 | 17.09.2024 |  |
| 2 | Описание и анализ предметной области | 10 | 24.09.2024 |  |
| 3 | Проектирование системы | 40 | 13.12.2024 |  |
| 3.1 | Разработка структурной схемы системы | 5 | 8.10.2024 |  |
| 3.2 | Разработка функциональной спецификации системы и прототипа интерфейса пользователя | 10 | 25.10.2024 |  |
| 3.3 | Разработка информационно-логического проекта системы и его предъявление руководителю | 25 | 13.12.2024 |  |
| 4 | Реализация проекта, разработка контрольных примеров. Предъявление реализации руководителю | 40 | 13.12.2024 |  |
| 5 | Корректировка проекта и оформление документации проекта. Защита проекта с представлением презентации. | 5 | 27.12.2024 |  |

Задание принял  
 к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.А. Атякшев

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.О. Колбанов

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р.К. Лукашевич

ПРИЛОЖЕНИЕ  
к заданию на курсовую работу  
обучающимся в группе № 6401-020302D  
С.А. Атякшеву

Д.О. Колбанову

Р.К. Лукашевичу

Тема проекта: «Клавиатурный тренажёр с функциями администратора»

Исходные данные к проекту:

1. Характеристика объекта автоматизации:

## объект автоматизации: клавиатурный тренажёр;

## виды автоматизируемой деятельности:

* + процесс авторизации/регистрации пользователей;
  + процесс ручного составления упражнения;
  + процесс генерации упражнения;
  + процесс выполнения упражнения;
  + процесс настройки уровня сложности;
  + процесс визуализации работы клавиатуры;

## количество ролей пользователей – 2;

## минимальная длина логина – 4 символа;

## максимальная длина логина – 8 символов;

## минимальная длина пароля – 4 символа;

## максимальная длина пароля – 10 символов;

## минимальное количество знаков в упражнении – 25;

## максимальное количество знаков в упражнении – 200;

## минимальное время нажатия на клавишу – 0,3 с;

## максимальное время нажатия на клавишу – 1,5 с;

## максимальное количество допустимых ошибок – 5;

## количество уровней сложности – 5;

## количество зон клавиатуры – 5;

1. количество способов создания упражнения – 2;
2. количество видов статистики – 2;
3. количество видов отображения статистики – 3.
4. Требования к информационному обеспечению:
5. информационное обеспечение разрабатывается на основе следующих источников:
   * клавиатурный тренажёр [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Keyboard\_trainer (дата обращения: 14.09.2024);
6. звуковые эффекты хранятся в файлах формата wav;
7. структура базы данных (БД) разрабатывается на основании следующих сведений:
   * об игроке (логин, пароль, роль, статистика);
   * об упражнении (номер, текст упражнения, уровень сложности);
   * о статистике (среднее время нажатий, количество пройденных уровней);
   * об уровне сложности (номер уровня, зоны клавиатуры, время нажатия на клавишу, количество допустимых ошибок, минимальное количество знаков, максимальное количество знаков).
8. должна быть обеспечена целостность базы данных и защита от несанкционированного доступа.
9. Требования к техническому обеспечению:
   1. Требования к техническому обеспечению серверной части:
10. тип ЭВМ – IBM PC совместимый;
11. объем ОЗУ – не менее 2 Гб;
12. объем свободного пространства на внешнем диске – не менее 50 Гб;
13. наличие подключения к сети Интернет;
14. манипулятор – мышь;
15. технические характеристики определяются в процессе выполнения проекта.
    1. Требования к техническому обеспечению клиентской части:
16. тип ЭВМ – IBM PC совместимый;
17. монитор с разрешающей способностью не ниже 800 х 600;
18. манипуляторы – клавиатура и мышь;
19. технические характеристики определяются в процессе выполнения проекта.
20. Требования к программному обеспечению:
    1. Требования к программному обеспечению серверной части:
21. тип операционной системы – Windows 7 и выше;
22. СУБД – PostgreSQL 17.
    1. Требования к программному обеспечению клиентской части:
23. тип операционной системы – Windows 7 и выше;
24. браузер – Google Chrome 86.0.4240.183 (64-битный) и выше, Firefox 83.0 (64-битный) и выше.
    1. Требования к программному обеспечению рабочего места разработчика:
25. тип операционной системы – Windows 7 и выше;
26. язык программирования – TypeScript, Rust HTML, CSS;
27. среда программирования – Visual Studio Code;
28. СУБД – PostgrateSQL 17;
29. фреймворк – React, Axum;
30. среда проектирования – StarUML 6.2.2.
31. Общие требования к проектируемой системе:

5.1 Функции, реализуемые системой:

1. функции системы:
   * аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
   * автоматическое составление текста упражнения по заданным параметрам;
   * формирование списка упражнений для заданного уровня сложности;
   * генерация длины упражнения;
   * вычисление средней скорости набора;
   * формирование статистики;
   * контроль набора введённых символов;
   * визуализация процесса работы с клавиатурой;
   * контроль количества допущенных ошибок;
   * выдача сообщений пользователю;
   * выдача справочной информации о системе;
2. функции администратора:
   * авторизация пользователя в системе (ввод логина и пароля);
   * настройка уровня сложности:
3. ввод номера уровня сложности;
4. задание минимального количества символов;
5. задание максимального количества символов;
6. выбор зон клавиатуры;
7. задание количества допустимых ошибок;
   * создание/редактирование упражнения:
8. выбор уровня сложности;
9. выбор способа создания упражнения;
10. задание длины упражнения (для автоматического режима создания упражнения);
11. ручной ввод текста упражнения;
    * сохранение упражнения в БД;
    * загрузка упражнения из БД;
    * просмотр справочной информации;
    * работа со статистикой:
12. по пользователям;
13. по упражнениям;
14. функции игрока:
    * регистрация пользователя в системе (ввод логина, пароля, имени);
    * авторизация пользователя в системе (ввод логина и пароля);
    * выбор уровня сложности;
    * выбор упражнения из списка;
    * выполнение упражнения;
    * просмотр собственной статистики;
    * включение/выключение виртуальной клавиатуры;
    * включение/выключение звуковых эффектов;
    * просмотр справочной информации.

5.2 Технические требования к системе:

1. режим работы – диалоговый;
2. время автоматической генерации упражнения – не более 10 с;
3. система должна удовлетворять санитарным правилам и нормам  
    СанПин 2.2.2./2.4.2198-07;
4. условия работы средств вычислительной техники (содержание вредных веществ, пыли и подвижность воздуха) должны соответствовать ГОСТ 12.1.005, 12.01.007;
5. температура окружающего воздуха – 15-35°С;
6. влажность воздуха – 45-75%.

Руководитель   
проекта Л.С. Зеленко

Задание принял  
к исполнению С.А. Атякшев

Д.О. Колбанов

Р.К. Лукашевич

РЕФЕРАТ

Пояснительная записка 40 с, 14 рисунков, 5 таблиц[[1]](#footnote-1), 12 источников,  
2 приложения.

Графическая часть: ??? слайдов презентации PowerPoint.

ДЕРЕВО ПОИСКА, ГЕНЕРАТОР КРОССВОРДОВ, ГОЛОВОЛОМКА, СЛОВАРЬ ТЕРМИНОВ, ВАРИАНТ ОТОБРАЖЕНИЯ, РАЗГАДЫВАНИЕ

Объектом автоматизации является клавиатурный тренажёр.

Во время курсового проектирования разработаны алгоритмы и соответствующая им программа, позволяющая выполнять автоматическую генерацию линейного кроссворда по заданной теме. Задания (понятие и его расшифровка) хранятся в текстовом файле и могут дополняться вручную внутри программы, при этом ограничений на длину словаря не существует. Тема кроссворда выбирается пользователем в соответствии с содержанием словаря заданий. Программа позволяет сформировать кроссворд, учитывая ограничения на параметры. В системе имеется возможность сохранения кроссвордов в файл с целью последующего их разгадывания.

Программа написана на языке С# в среде Visual Studio 2015 и функционирует под управлением операционной системы Windows 7 и выше. Доступ к данным осуществляется с помощью СУБД PostgrateSQL 17.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 12](#_Toc178663297)

[1 Описание и анализ предметной области 14](#_Toc178663298)

[1.1 Описание предметной области 14](#_Toc178663299)

[1.2 Описание систем-аналогов 20](#_Toc178663300)

[1.2.1 Stamina-online 20](#_Toc178663301)

[1.2.2 Klavarog 21](#_Toc178663302)

[1.3 Диаграмма объектов предметной области 23](#_Toc178663303)

[1.4 Постановка задачи 24](#_Toc178663304)

[2 Проектирование системы 28](#_Toc178663305)

[2.1 Выбор и обоснование архитектуры системы 28](#_Toc178663306)

[2.2 Структурная схема системы 28](#_Toc178663307)

[2.3 Разработка спецификации требований 32](#_Toc178663308)

[2.3.1 Функциональная спецификация 32](#_Toc178663309)

[2.3.2 Перечень исключительных ситуаций 32](#_Toc178663310)

[2.4 Разработка прототипа интерфейса пользователя системы 35](#_Toc178663311)

[2.5 Разработка информационно-логического проекта системы 36](#_Toc178663312)

[2.5.3 Язык UML 38](#_Toc178663313)

[2.5.4 Диаграмма вариантов использования 38](#_Toc178663314)

[2.5.5 Сценарии 38](#_Toc178663315)

[2.5.6 Диаграмма классов 40](#_Toc178663316)

[2.5.7 Диаграмма состояний 40](#_Toc178663317)

[2.5.8 Диаграмма деятельности 40](#_Toc178663318)

[2.5.9 Диаграмма последовательности 41](#_Toc178663319)

[2.6 Логическая модель данных (при необходимости) 42](#_Toc178663320)

[2.7 Выбор и обоснование алгоритмов обработки данных /Разработка и описание алгоритмов обработки данных 43](#_Toc178663321)

[2.8 Выбор и обоснование комплекса программных средств 45](#_Toc178663322)

[2.8.1 Выбор языка программирования 45](#_Toc178663323)

[2.8.2 Выбор среды программирования 46](#_Toc178663324)

[2.8.3 Выбор операционной системы 46](#_Toc178663325)

[2.8.4 Выбор системы управления базами данных (при необходимости) 46](#_Toc178663326)

[3 Реализация системы 47](#_Toc178663327)

[3.1 Разработка и описание интерфейса пользователя 47](#_Toc178663328)

[3.2 Диаграммы реализации 48](#_Toc178663329)

[3.2.1 Диаграмма компонентов 48](#_Toc178663330)

[3.2.2 Диаграмма развертывания 49](#_Toc178663331)

[3.2.3 Диаграмма классов 49](#_Toc178663332)

[3.3 Физическая модель данных (при необходимости) 50](#_Toc178663333)

[3.4 Выбор и обоснование комплекса технических средств 51](#_Toc178663334)

[3.4.1 Расчет объема занимаемой памяти 51](#_Toc178663335)

[3.4.2 Минимальные требования, предъявляемые к системе 53](#_Toc178663336)

[ЗАКЛЮЧЕНИЕ 54](#_Toc178663337)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 55](#_Toc178663338)

[ПРИЛОЖЕНИЕ А Руководство пользователя 58](#_Toc178663339)

[А.1 Назначение системы 58](#_Toc178663340)

[А.2 Условия работы системы 58](#_Toc178663341)

[А.3 Установка системы 58](#_Toc178663342)

[А.4 Работа с системой 59](#_Toc178663343)

[А.4.1 Работа с системой в режиме администратора (если необходимо) 59](#_Toc178663344)

[А.4.2 Работа с системой в режиме пользователя 59](#_Toc178663345)

[ПРИЛОЖЕНИЕ Б Листинг модулей программы 60](#_Toc178663346)

ВВЕДЕНИЕ

Клавиатурный тренажер – это инструмент, который помогает пользователям улучшить скорость и точность печати на клавиатуре. Несмотря на то, что сегодня клавиатура стала неотъемлемой частью нашей жизни, её история насчитывает не один десяток лет.

Первые попытки создать системы обучения печати относятся к концу 19-го века. В 1888 году Франк Эдвард МакГуррин издал книгу "Touch Typewriting", которая стала первым учебником по слепой печати.

В начале 20-го века появились механические клавиатурные тренажеры, которые использовали систему с "закрепленными" клавишами, чтобы научить пользователей правильно позиционировать пальцы.

С развитием компьютеров появились электронные клавиатурные тренажеры, которые могли отслеживать скорость и точность печати, а также предлагать упражнения и тесты. Mavis Beacon Teaches Typing – одна из самых известных ранних программ для обучения набору текста. Она предлагала пользователям интерактивные уроки, отслеживала прогресс и давала рекомендации по улучшению техники. Это была одна из первых программ, которая использовала игровой подход, превращая обучение в интересный процесс. В конце 1980-х и 1990-х годах стали популярны обучающие программы, которые использовались в образовательных учреждениях для тренировки печати у школьников и студентов [1].

Во время курсового проектирования необходимо разработать клавиатурный тренажёр, с помощью которого можно будет выполнять упражнения для тренировки слепой печати, также с помощью данной системы можно будет конструировать упражнения в ручном или автоматическом режиме в соответствии с заданным уровнем сложности.

Разработка системы будет производиться по технологии быстрой разработки приложений RAD (Rapid Application Development)*,* которая поддерживается методологией структурного проектирования и включает элементы объектно-ориентированного проектирования и анализа предметной области [2].

При проектировании системы будут использоваться методология ООАП (Object-Oriented Analysis/Design), в основу которой положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов, и язык моделирования UML (Unified Modeling Language), который является стандартным инструментом для разработки «чертежей» программного обеспечения [3].

1. Описание и анализ предметной области

Под предметной областью (application domain) принято понимать ту часть реального мира, которая имеет существенное значение или непосредственное отношение к процессу функционирования программы. Другими словами, предметная область включает в себя только те объекты и взаимосвязи между ними, которые необходимы для описания требований и условий решения некоторой задачи [4].

* 1. Описание предметной области

Клавиатурный тренажёр – вид компьютерных программ или онлайн-сервисов, предназначенных для обучения набору на компьютерной клавиатуре [?].

Компьютерная клавиатура – устройство ввода. Представляет собой набор клавиш (кнопок), расположенных в определённом порядке. Бывает совмещённой в одно устройство с тачпадом, тензометрическим джойстиком или трекболом. На рисунке 1 представлен пример компьютерной клавиатуры.

  
Рисунок 1 – Пример компьютерной клавиатуры

Обычно целями тренажёров являются:

* научить слепому методу печати, в частности, задействовать для набора все десять пальцев рук;
* увеличить скорость набора;
* уменьшить количество опечаток;
* улучшить ритмичность набора (что позволяет уменьшить усталость при наборе).

Существуют несколько типов клавиатур, далее будут рассмотрены наиболее распространённые из них.

Мембранные клавиатуры являются самыми многочисленными и дешёвыми на рынке клавиатур. Мембранная клавиатура состоит из двух основных слоев: нижнего и верхнего. Верхний слой содержит клавиши с символами, а нижний слой содержит металлические контакты. Когда клавиша нажимается, верхний слой сжимается и соприкасается с нижним слоем, что приводит к замыканию контакта и передаче сигнала на компьютер. На рисунке 3 представлен пример мембранной клавиатуры.

  
Рисунок 3 – Пример мембранной клавиатуры

Механические клавиатуры используют переключатели. Когда клавиша нажимается, переключатель срабатывает и контакты замыкаются, что приводит к передаче сигнала на компьютер. Механические клавиатуры также имеют разные типы переключателей, такие как линейные, тактильные и кликовые, каждый из которых имеет свои уникальные характеристики и ощущения при нажатии клавиш. Также от них зависит громкость нажатия [5]. На рисунке 4 представлен пример механической клавиатуры.

  
Рисунок 4 – Пример механической клавиатуры

Также существуют несколько раскладок клавиатуры. Раскладка клавиатуры – соглашение о соответствии типографических символов (букв, цифр, знаков препинания и т. д.) письменного языка клавишам клавиатуры компьютера, пишущей машинки или другого устройства, с помощью которого вводится текст [5]. Среди раскладок для английского языка выделяют QWERTY и Colemak. При создании раскладки QWERTY требовалось, чтобы буквы, образующие в английском языке устойчивые комбинации, располагались как можно дальше друг от друга по разные стороны клавиатуры и были разбросаны по разным рядам, что уменьшало вероятность «перепутывания» рычажков пишущей машинки. В 2006 году Шаем Коулманом (Shai Coleman) была разработана раскладка Colemak. Название происходит от Coleman+Dvorak. Раскладка приспособлена к современным компьютерным реалиям. Её принцип — эффективный и эргономичный набор текстов на английском языке на компьютерной клавиатуре. Примеры раскладок QWERTY и Colemak представлены на рисунках 5 и 6 соответственно.

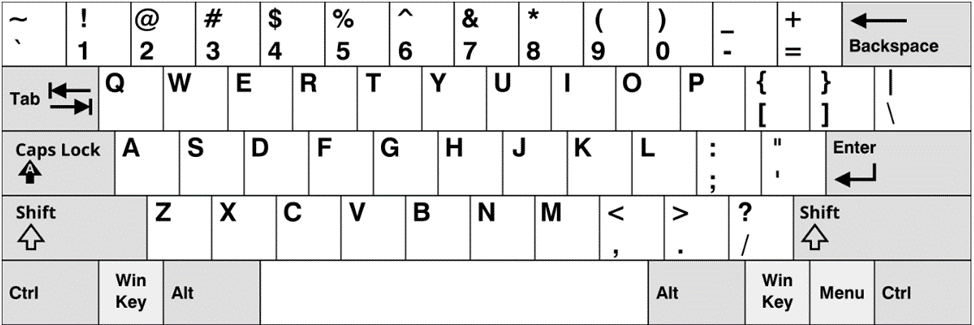
  
Рисунок 5 – Пример раскладки QWERTY

  
Рисунок 6 – Пример раскладки Colemak

Среди русских раскладок клавиатуры выделяют ЙЦУКЕН и «фонетическая раскладку». Более распространённой из них является раскладка ЙЦУКЕН, название которой происходит от шести левых символов верхнего ряда раскладки. Считается, что раскладка ЙЦУКЕН далека от оптимальной при печати слепым десятипальцевым методом [6]:

* нагрузка по пальцам распределяется неравномерно, из-за чего одни пальцы «пробегают» по клавиатуре большие расстояния, чем другие;
* приходится часто гнуть пальцы, так как основной «домашний» ряд клавиатуры мало задействован;
* часто два и более символа нажимаются подряд в одной зоне, одним пальцем.

Пример раскладки ЙЦУКЕН представлен на рисунке 7.

  
Рисунок 7 – Пример раскладки ЙЦУКЕН

Чтобы быстро выполнить запись текстового материала следует овладеть слепым методом печати, когда каждой клавише с буквой или цифрой соответствует свой палец. На рисунке 8 представлены зоны клавиатуры для каждого пальца.

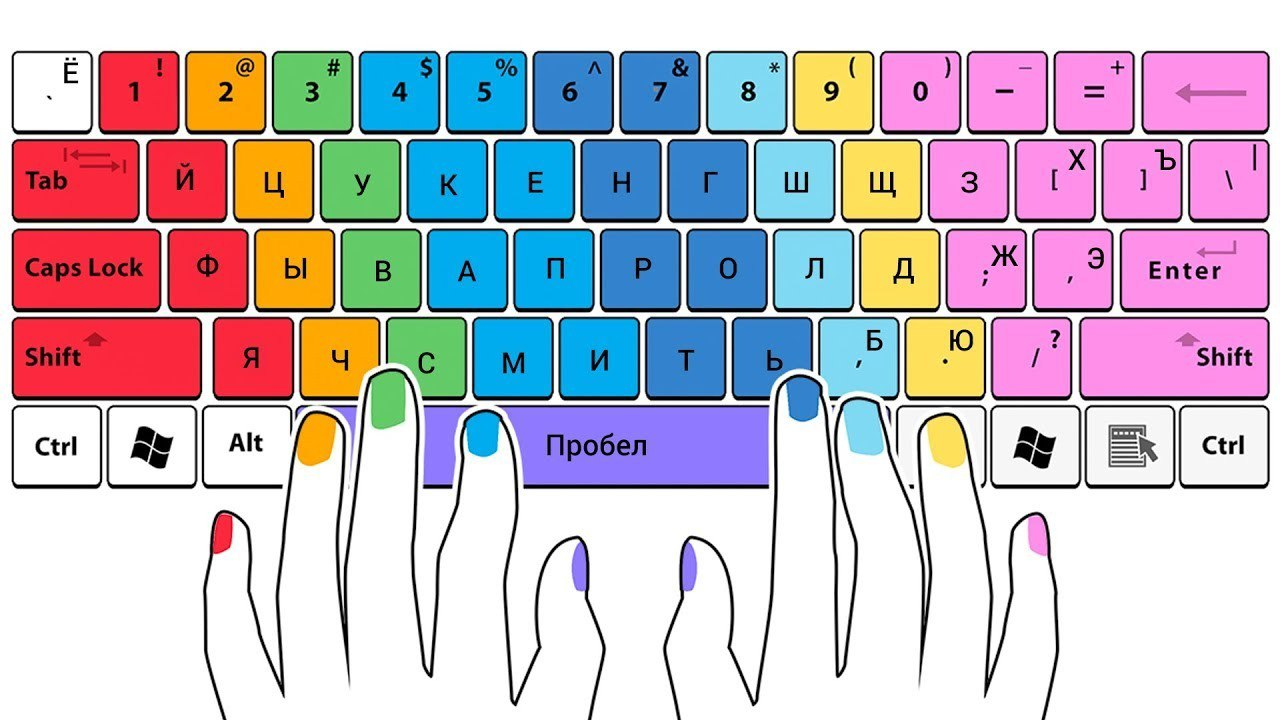


Рисунок 8 – Зоны клавиатуры

Такое расположение букв на клавиатуре закономерно. Те буквы, которые используются чаще всего в текстах, сосредоточены в центре, и за них отвечают указательные пальцы – самые подвижные и натренированные. Редкие буквы и клавиши, находящиеся рядом с буквенной клавиатурой, отводятся мизинцам [7].

В таблице 1 представлены символы разных зон клавиатуры.

Таблица 1 – Зоны клавиатуры

|  |  |
| --- | --- |
| Зоны клавиатуры | Набор символов |
| Синяя, темно-синяя | 4, 5, 6, 7, к, е, н, г, а, п, р, о, м, и, т, ь |
| Зеленая, голубая | 3, 8, у, ш, в, л, с, б |
| Оранжевая, желтая | 2 ,9, ц, щ, ы, д, ч, ю |
| Красная, розовая | 1, 0, й, з, х, ъ, ф, ж, э, я |
| Фиолетовая | Пробел |

Методика составления упражнений соответствует принципам методики В.В. Шахиджаняна. Методика Шахиджаняна является одной из самых эффективных методик обучения слепой печати. Она помогает научиться печатать быстро и точно, а также развивает координацию пальцев и увеличивает скорость реакции. Основные принципы методики Шахиджаняна [8]:

* постепенное усложнение: упражнения становятся все более сложными по мере освоения навыков;
* повторение: упражнения повторяются несколько раз, чтобы закрепить навыки;
* индивидуальный подход: методика предусматривает возможность изменения упражнений в зависимости от индивидуальных особенностей обучающегося.
  1. Описание систем-аналогов

В интернете существуют большое количество версий клавиатурного тренажёра. Все они разные имеют свои особенности и функционал. Рассмотрим основные характеристики нескольких систем-аналогов.

* + 1. Stamina-online

Тренажер Stamina-online помогает освоить десятипальцевую слепую машинопись не только традиционным методом расположения пальцев на клавишах ASDF, но и альтернативным – когда пальцы лежат на нижнем ряду клавиатуры SDFV [8]. По мнению разработчика, в таком положении кисти рук меньше устают. На сегодняшний день это единственный тренажер, который позволяет отработать ввод из двух разных исходных позиций. Из практических занятий доступны уроки для запоминания расположения клавиш и работа с текстами/фразами, предоставляемыми как программой, так и из любого внешнего файла. Интересным образом выполнена виртуальная клавиатура: она не только подсвечивает нужные буквы, но и примерную область работы каждого пальца. Своеобразной «фишкой» программы является легкий и юморной язык, которым написаны FAQ и разделы помощи для пользователя: полезная информация разбавляется анекдотами и интересными логическими задачами. На рисунке 4 приведена страница сайта «Stamina-online».

  
Рисунок 9 – Страница сайта «Stamina-online»

К достоинствам данной системы относятся:

* бесплатный доступ;
* понятный интерфейс;
* наличие обучения из альтернативного варианта расположения пальцев;
* отображение используемых пальцев;
* наличие теории;
* возможность тренироваться на языках программирования;
* визуализация клавиатуры.

К недостаткам системы относятся:

* большая нагрузка на руки с первых занятий;
* большое количество рекламы;
* отсутствие визуализации статистики.
  + 1. Klavarog

Клавиатурный тренажёр «Klavarog» удобен тем, что предлагает несколько уровней сложности и отдельную категорию упражнений на скорость: на «начальном» можно освоить первичные навыки десятипальцевой печати классическим методом и только потом продвигаться дальше в удобном для себя темпе [9]. Из словаря подбираются слова, которые нужно набрать без ошибок, но в случае неоднократных опечаток тренажер предлагает ввести проблемные слова заново. На сайте есть таймер, но его основная задача – это организация отдыха и перерывов, а не «наказание» пользователя за несоблюдение минимума набранных знаков в минуту. В разделе статистики собирается информация о всех сессиях, скорости (по количеству символов в минуту и реальному числу введенных слов) и ошибках. А еще у сайта есть приятная опция «Дзен», которую можно включить нажатием на логотип – в этом режиме можно просто расслабиться во время печати 10.

К достоинствам данной системы относятся:

* бесплатный доступ;
* визуализация клавиатуры;
* возможность тренироваться на языках программирования.

К недостаткам системы относятся:

* отсутствие обновлений;
* отсутствие звукового сопровождения;
* отсутствие уровней сложности;
* устаревший интерфейс;
* отсутствие статистики.

  
Рисунок 10 – Страница сайта «Klavarog»

На основании анализа возможностей систем-аналогов были сформулированы требования к разрабатываемой системе (см. таблицу 2).

Таблица 2 – Сравнительные характеристики систем-аналогов

|  |  |  |  |
| --- | --- | --- | --- |
| Название системы  Название показателя | Система «Stamina-online» | Система «Klavarog» | Разрабатываемая система |
| Бесплатный доступ | + | + | + |
| Удобство использования | + | - | + |
| Уровень сложности | + | - | + |
| Звуковое сопровождение | + | - | + |
| Визуализация клавиатуры | + | + | + |
| Визуализация статистики | - | - | + |
| Упражнения на иностранные языки | + | + | - |
| Подсчёт времени | + | + | + |

* 1. Диаграмма объектов предметной области

В объектном или объектно-ориентированном подходе в первую очередь выделяется множество основных объектов системы и впоследствии определяется множество операций над объектами. Такой подход базируется на абстрактных типах, и решение задачи выражается в терминах выделенных объектов [10].

Объектно-ориентированный анализ и проектирование (ООАП, Object-Oriented Analysis/Design) - технология разработки программных систем, в основу которых положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов [11].

На рисунке 12 приведена диаграмма объектов предметной области.

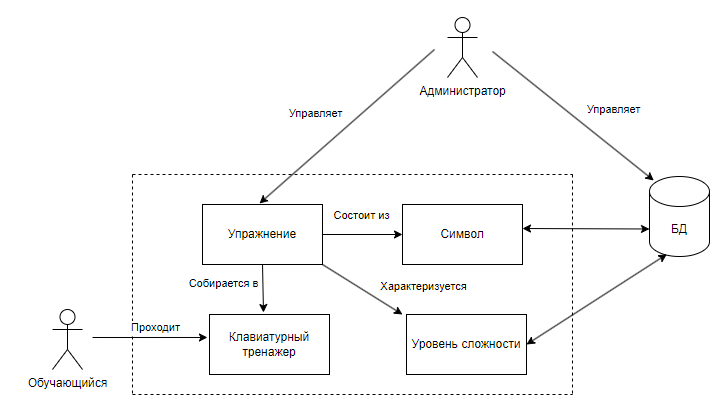


Рисунок 12 – Диаграмма объектов предметной области

Как видно из диаграммы, клавиатурный тренажер представляет собой совокупность упражнений, которые в свою очередь состоят из символов и характеризуются уровнем сложности, который определяет количество допустимых ошибок, максимальную и минимальную скорость нажатия и количество символов. Упражнения создаются и редактируются администратором, который также управляет базой данных.

* 1. Постановка задачи

Во время курсового проектирования необходимо разработать автоматизированную систему составления и прохождения упражнений для тренировки слепой печати, с помощью которой администратор может конструировать упражнение в ручном или автоматическом режиме в соответствии с заданным уровнем сложности, а обучающийся просматривать статистику по упражнениям. Система должна быть реализована в виде веб-приложения.

В системе должно быть реализовано две роли пользователей: администратор и игрок, поэтому для того, чтобы работать с системой, при первом входе в систему пользователи должны пройти процедуру регистрации: ввести логин и пароль (длина логина должна быть от 4 до 8 символов, длина пароля – от 4 до 10 символов). При повторном входе, пользователю необходимо авторизоваться в системе и затем, после аутентификации, система должна настроить интерфейс пользователя на заданную роль.

Администратору в первую очередь должна быть доступна функция создания и редактирования упражнения. Администратор должен иметь возможность создавать и редактировать упражнения, а именно выбирать уровень сложности (от 1-ого до 5-ого), вручную вводить текст упражнения (от 25 до 200 символов), выбирать способ создания упражнения (ручной или автоматический), задавать длину упражнения (для автоматического режима создания упражнения).

Также администратору должна быть доступна функция настройка уровня сложности. Администратор должен иметь возможность настраивать уровень сложности, который будет зависеть от заданного минимального и максимального количества символов, зоны клавиатуры и максимального количества допустимых ошибок.

Функции авторизации пользователя в системе и работы со статистикой (по пользователям и по упражнениям) также должны быть доступны администратору;

Основная функция игрока – это выполнение упражнений. Игрок должен выбрать упражнение, которое характеризуется уровнем сложности (в системе должно быть реализовано 5 уровней сложности) и количеством знаков. Также система должна быть обеспечена звуковым сопровождением, которое пользователь, при желании, сможет выключить, и визуализацией клавиатуры.

У пользователя должна быть возможность посмотреть статистику по упражнению. В системе также должна быть обеспечена возможность получения справочной информации как о самой системе, так и предоставляемых ею возможностях.

Таким образом, система должна решать следующие задачи:

1. функции системы:
   * аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
   * автоматическое составление текста упражнения по заданным параметрам;
   * формирование списка упражнений для заданного уровня сложности;
   * генерация длины упражнения;
   * вычисление средней скорости набора;
   * формирование статистики;
   * контроль набора введённых символов;
   * визуализация процесса работы с клавиатурой;
   * контроль количества допущенных ошибок;
   * выдача сообщений пользователю;
   * выдача справочной информации о системе;
2. функции администратора:
   * авторизация пользователя в системе (ввод логина и пароля);
   * настройка уровня сложности:
3. ввод номера уровня сложности;
4. задание минимального количества символов;
5. задание максимального количества символов;
6. выбор зон клавиатуры;
7. задание количества допустимых ошибок;
   * создание/редактирование упражнения:
8. выбор уровня сложности;
9. выбор способа создания упражнения;
10. задание длины упражнения (для автоматического режима создания упражнения);
11. ручной ввод текста упражнения;
    * сохранение упражнения в БД;
    * загрузка упражнения из БД;
    * просмотр справочной информации;
    * работа со статистикой:
12. по пользователям;
13. по упражнениям;
14. функции игрока:
    * регистрация пользователя в системе (ввод логина, пароля, имени);
    * авторизация пользователя в системе (ввод логина и пароля);
    * выбор уровня сложности;
    * выбор упражнения из списка;
    * выполнение упражнения;
    * просмотр собственной статистики;
    * включение/выключение виртуальной клавиатуры;
    * включение/выключение звуковых эффектов;
    * просмотр справочной информации.
15. Проектирование системы
    1. Выбор и обоснование архитектуры системы

Архитектура – принципиальная организация системы, воплощенная в её элементах, их взаимоотношениях друг с другом и со средой, а также принципы, направляющие её проектирование и эволюцию [12].

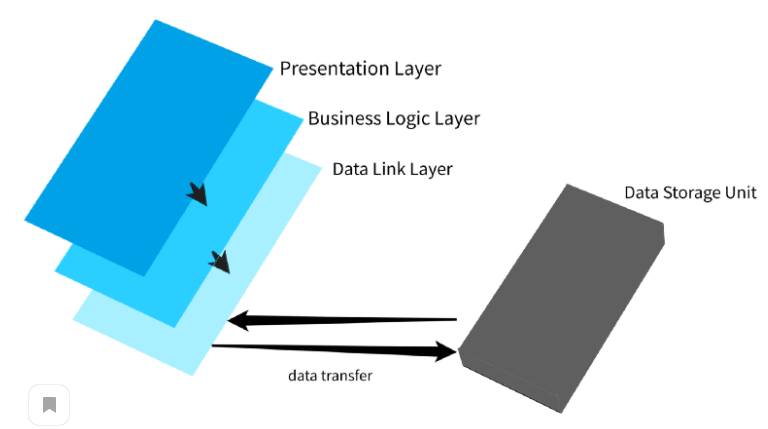
* + 1. Виды архитектур

За годы развития ПО разработчикам удалось придумать надёжные подходы, чтобы устранить недостатки проектирования без архитектуры [13].

Рассмотрим следующие виды архитектур.

Многослойная архитектура

Многослойная архитектура (n-tier client-server architecture) – архитектура, где процессы представления, обработки и управления данными являются логически отделенными друг от друга процессами (см. рисунок 13). Модель многослойной архитектуры помогает создать гибкое и многократно используемое программное обеспечение. В случае изменений надо их делать лишь в отдельных слоях, а не сразу во всем приложении. Это сулит меньше работы, меньших затрат времени и меньше потенциальных ошибок.

  
Рисунок 11 – Многослойная архитектура

Более типичным и более используемым вариантом является трехслойная архитектура (three-tier client server architecture). В случае трехслойного приложения каждый слой располагается в разных местах в компьютерной сети и может располагаться также ​​на разных платформах.

К пользователю ближе всего находящееся в его компьютере программное обеспечение рабочей станции (так называемый слой логики представления). Этот слой может отводиться формам ввода и типичному на данной платформе графическому пользовательскому интерфейсу. Не исключено существование этого слоя на различных платформах. Слой логики представления взаимодействует со слоем логики приложения (также слой бизнес-логики, средний слой).

Задача бизнес-логики или слоя логики фирмы (компании) - управлять функциональностью, обрабатывая для этого полученные с нижнего слоя данные в соответствии с запросами пользователя, пришедшими с верхнего слоя. Этот слой обычно находится на сервере локальной сети.

Слой данных - это третий слой. Он включает в себя базу данных и необходимое для ее управления программное обеспечение и может располагаться на некотором мейнфрейме. Слой отвечает среди прочего за сохранение данных независимо от приложения и логики представления.

В настоящее время у широкораспространенных веб-приложений, как правило, аналогичная архитектура. В этом случае слой логики представления связан с веб-браузером (веб-обозревателем) и с визуализацией пользовательского интерфейса (рендеринг) в окне браузера.

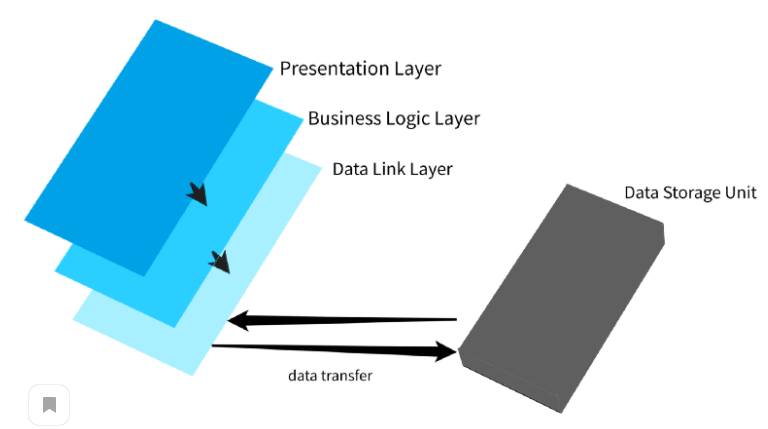
Часть содержания может быть статической и часть - динамической.

Обобщением является многослойная архитектура (n-tier architcture). Здесь различные слои могут добавляться по мере необходимости (или описанные слои могут быть разделены, в свою очередь, на части). В ситуации, когда все большая ценность имеется не у одиночных приложений или баз данных, а у взаимодействующих информационных систем, узкое место в разработке переместилось в сторону создания интерфейсов и интеграции систем. Немаловажно использовать проверенные практики (например, шаблоны для разрешения аналогичных проблем), сервис-ориентированный подход, семантическое описание информационных систем.

Этот подход работает по принципу разделения ответственностей. ПО разделено на слои (см. рисунок 11), лежащие друг на друге, и каждый из них выполняет определённую обязанность.

Архитектура делит ПО на следующие слои:

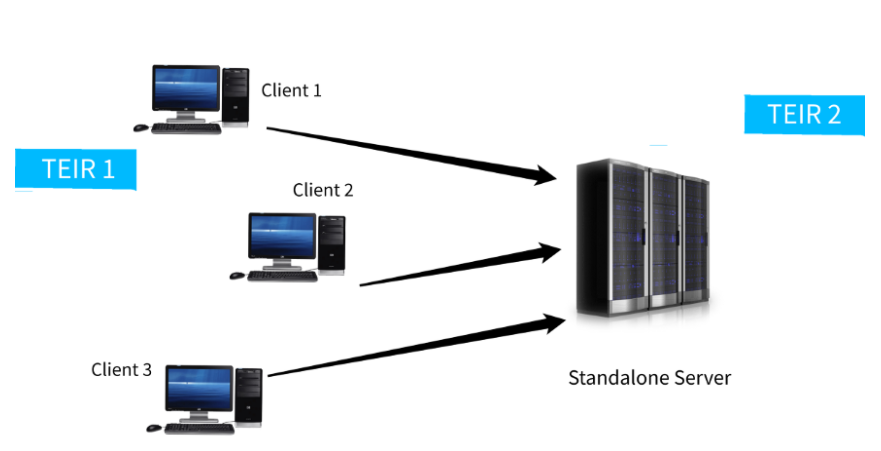
* Слой представления (Presentation Layer) содержит пользовательский интерфейс и отвечает за обеспечение хорошего пользовательского опыта.
* Слой бизнес-логики (Business Logic Layer), как следует из названия, содержит бизнес-логику приложения. Он отделяет UI/UX от вычислений, связанных с бизнесом. Это позволяет с лёгкостью изменять логику в зависимости от постоянно меняющихся бизнес-требований, никак не влияя на другие слои.
* Слой передачи данных (Data Link Layer) отвечает за взаимодействие с постоянными хранилищами, такими как базы данных, и прочую обработку информации, которая не связана с бизнесом.

  
Рисунок 11 – Многослойная архитектура

Данные и элементы управления проходят через каждый слой в дизайне и передаются от одного к другому. Эта система также повышает уровень абстракции и в некоторой степени даже стабильность ПО.

Двухуровневая система

Эта система состоит из двух физических машин в качестве сервера и клиента (см. рисунок 12). Она обеспечивает изоляцию операций управления данными, обработки данных и операций представления.

  
Рисунок 12 – Двухуровневая система

Клиент содержит слои презентации, бизнес-логики и передачи данных.

Сервер включает хранилища и базы данных.

N-уровневая система

Такие архитектуры обладают высокой масштабируемостью как по горизонтали, так и по вертикали. Реализация n-уровневой системы, как правило, обходится дороже, но обеспечивает высокую производительность. Поэтому она обычно применяется в крупных и комплексных программных решениях.

Сервис-ориентированная архитектура (SOA)

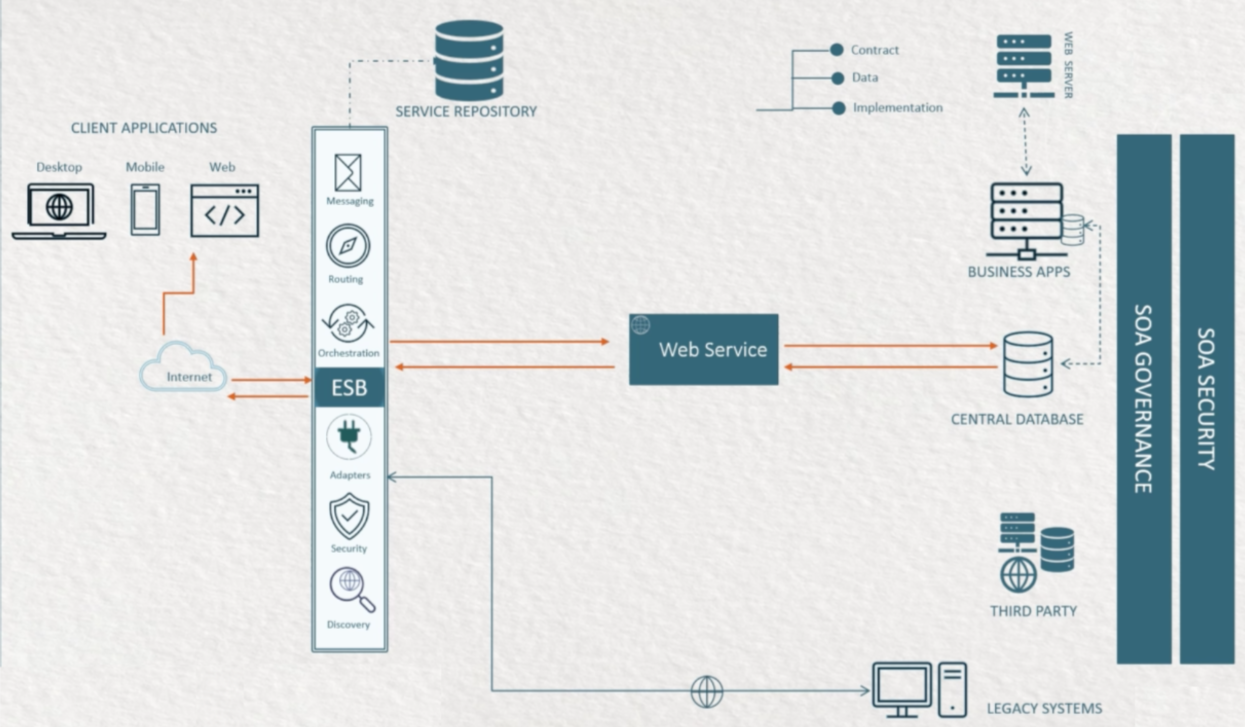
Эта архитектурная модель состоит из компонентов и приложений, которые связываются друг с другом с помощью чётко определённых сервисов (см. рисунок 13).

Она состоит из 5 элементов:

* сервисы (Services);
* сервисная шина (Service Bus);
* сервисный репозиторий (Service Repository catalogue of services);
* безопасность SOA (SOA Security);
* управление SOA (SOA Governance).

Клиент отправляет запрос с использованием стандартного протокола и формата данных по сети. Этот запрос обрабатывается ESB (сервисная шина предприятия), которая считается сердцем сервис-ориентированной архитектуры и отвечает за оркестровку и маршрутизацию. С помощью сервисного репозитория ESB направляет запрос в специальный сервис, который может взаимодействовать с другими сервисами и базами данных, чтобы составить полезную нагрузку (данные) ответа.

Полный вызов ответа на запрос согласуется с правилами управления и безопасности SOA для выполнения безопасной и корректной транзакции.

  
Рисунок 13 – Сервис-ориентированная архитектура (SOA)

Mикросервисная архитектура

При таком подходе приложение разрабатывается как набор небольших сервисов, каждый из которых работает в собственном процессе и связывается с легковесными механизмами, обычно API для HTTP-ресурса.

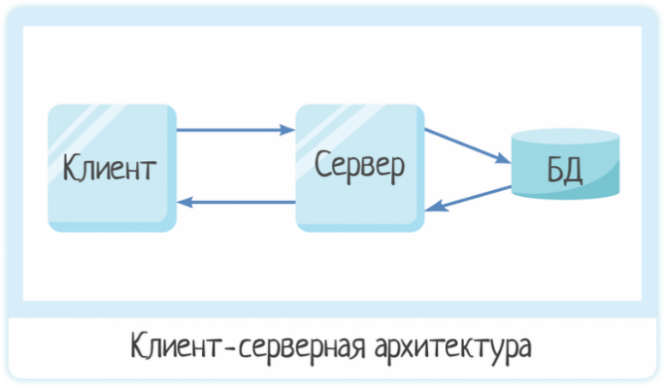
Эти сервисы основываются на бизнес-возможностях и могут развёртываться независимо друг от друга с помощью полностью автоматизированного механизма.

Централизованное управление между сервисами минимально. Они могут быть написаны на разных языках, использовать разные технологии хранения данных.

Архитектура работает по принципу компонентизации сервисов. Она разделяет программное обеспечение на различные изолированные компоненты (сервисы), каждый из которых несёт единую ответственность. Изменения в одной сервисе не должны затрагивать другие.

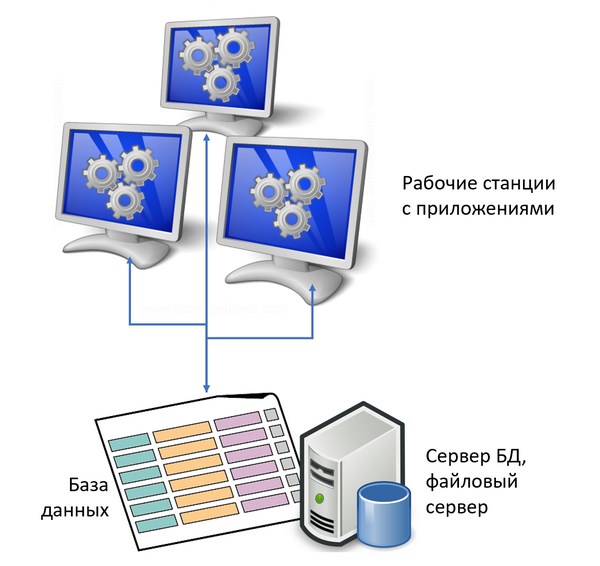
* + 1. Клиент-серверные архитектуры

Клиент-серверные архитектуры подразделяются по уровням. Схема клиент-серверной архитектуры представлена на рисунке 14.

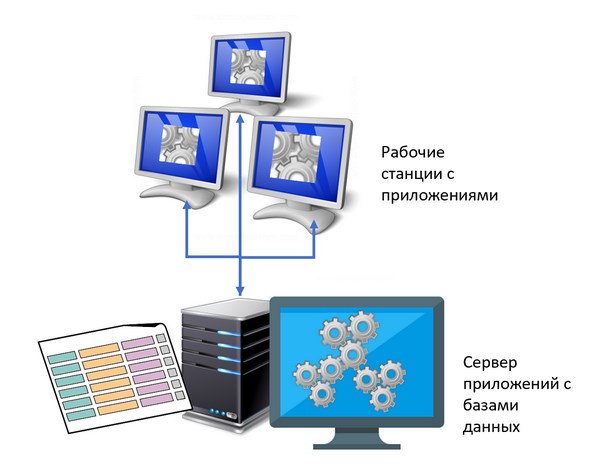
Рисунок 14 – Схема клиент-серверной архитектуры

Самыми используемыми являются следующие типы клиент-серверных архитектур [13]:

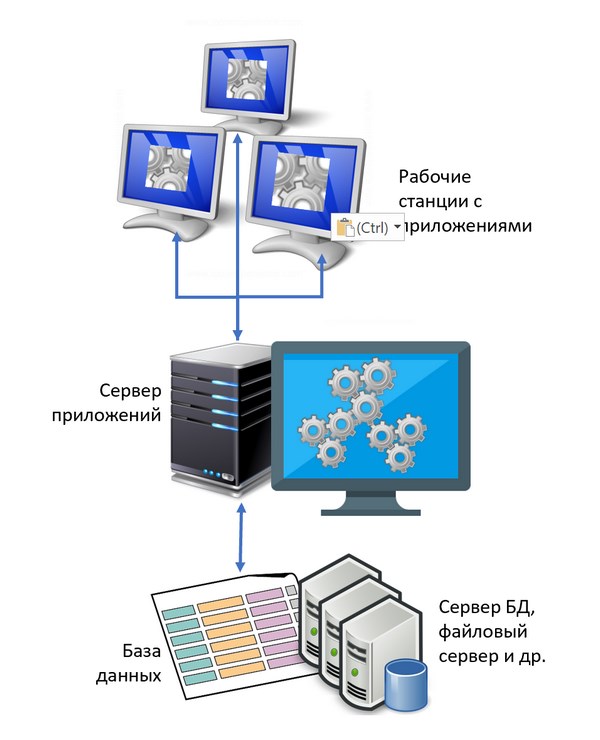
* Одноуровневая архитектура «клиент-сервер» (см. рисунок 15). В ней все прикладные программы рассредоточены по рабочим станциям, которые обращаются к общему серверу баз данных или к общему файловому серверу. Никаких прикладных программ сервер при этом не исполняет, только предоставляет данные.

  
Рисунок 15 – Одноуровневая архитектура

* К двухуровневой архитектуре (см. рисунок 16) «клиент-сервер» следует относить такую, в которой прикладные программы сосредоточены на сервере приложений (Application Server), а в рабочих станциях находятся программы-клиенты, которые предоставляют для пользователей интерфейс для работы с приложениями на общем сервере.

  
Рисунок 16 – Двухуровневая архитектура

* В трёхуровневой архитектуре (см. рисунок 17) сервер баз данных, файловый сервер и другие представляют собой отдельный уровень, результаты работы которого использует сервер приложений. Логика данных и бизнес-логика находятся в сервере приложений. Все обращения клиентов к базе данных происходят через промежуточное программное обеспечение (middleware), которое находится на сервере приложений. Вследствие этого, повышается гибкость работы и производительность.

  
Рисунок 17 – Трёхуровневая архитектура

* в многоуровневой архитектуре несколько серверов приложений используют результаты работы друг друга, а также данные от различных серверов баз данных, файловых серверов и других видов серверов.

Приложение будет реализовано на основе двухуровневой клиент-серверной архитектуре.

Преимущества двухуровневой архитектуры:

* легко конфигурировать и модифицировать приложения;
* пользователю обычно легко работать в такой среде;
* хорошая производительность и масштабируемость;

Однако, у двухуровневой архитектуры есть и ограничения:

* производительность может падать при увеличении числа пользователей;
* потенциальные проблемы с безопасностью, поскольку все данные и программы находятся на центральном сервере;
* все клиенты зависимы от базы данных одного производителя.
  + 1. Типы клиентов

Для двухуровневой архитектуры характерно три типа клиентов:

* Тонкий клиент – компьютер или программа-клиент в сетях с клиент-серверной или терминальной архитектурой, где большая часть задач по обработке информации перенесена на сервер и права доступа клиента строго ограничены.
* Толстый клиент – это приложение, обеспечивающее (в противовес тонкому клиенту) полную функциональность и независимость от центрального сервера. Часто сервер в этом случае является лишь хранилищем данных, а вся работа по обработке и представлению этих данных переносится на машину клиента.
* Веб-клиент – тип клиента, при котором все вычисления производятся на стороне сервера, а на стороне клиента не используется никаких прикладных программ, за исключением браузера.

В приложении будет использоваться тип толстого клиента. Все вычисления будут производиться на стороне клиента, на стороне сервера будет осуществляться хранение данных.

* + 1. Протоколы передачи данных

Протокол передачи данных – набор определённых правил или соглашений интерфейса логического уровня, который определяет обмен данными между различными программами. Эти правила задают единообразный способ передачи сообщений и обработки ошибок [14].

Различают множество различных протоколов, которые относятся к различным уровням в стеке TCP/IP. Некоторыми из самых известных являются [15]:

MAC – Media Access Control, низкоуровневый протокол, который используется для идентификации сетевых устройств.

IP – Internet Protocol, протокол который объединил различные устройства в единую сеть. Основная его задача – маршрутизация датаграмм. Данный протокол не устанавливает соединение, не подтверждает доставку пакетов получателю, как и не контролирует целостность данных.

TCP – Transmission Control Protocol, обеспечивает и контролирует передачу данных и следит за надёжностью и целостностью.

UDP – User Datagram Protocol, обеспечивает передачу данных, не создавая соединения. Этот протокол считается недостаточно надёжным, но имеет более высокую скорость доставки данных по сравнению с TCP

HTTP – Hyper Text Transfer protocol, обеспечивает передачу гипертекста, на его основе работают веб-сайты.

FTP – File Transfer Protocol, служит для передачи файлов.

SSH – Secure Shell. Протокол, отвечающий за создание удалённого канала для удаленного управления другой операционной системой.

Для реализации взаимодействия между сервером и клиентом устанавливаться соединение по протоколу TCP. Взаимодействие с СУБД, в свою очередь, будет осуществляться по протоколу mysql protocol, который передаёт данные в бинарном виде, и устанавливается «поверх» TCP соединения.

* 1. Структурная схема системы

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова. Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

Вводные слова про структурный подход к проектированию программных систем и комплексов программ + определение системы.

На рисунке ХХ приведена структурная схема разрабатываемой системы, в ее состав входят следующие подсистемы:

1. подсистема ???, которая отвечает за …;
2. подсистема ???, которая отвечает за …;
3. …

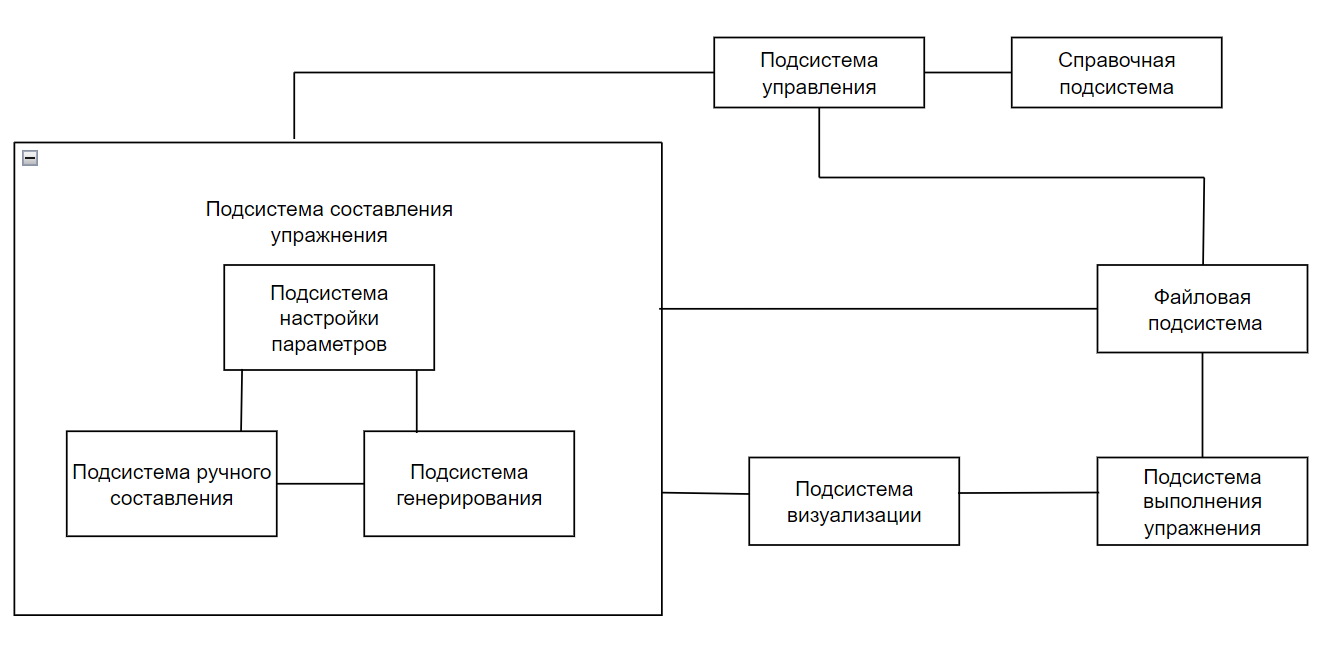
  
Рисунок ххх – Структурная схема системы (для АС составления и разгадывания ЛК)



Рисунок ххх – Структурная схема системы (для АС «Морской бой», технология «толстый клиент»)

  
Рисунок ххх – Структурная схема системы (для АС «Менеджмент», технология «тонкий» клиент»)

* 2. Разработка спецификации требований

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова. Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

* + 1. Функциональная спецификация

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Функциональная спецификация системы приведена в таблице 2.

* + 1. Перечень исключительных ситуаций

Исключительная ситуация – это ситуация, при которой система не может выполнить возложенных на нее функций или которая может привести к денормализации работы системы.

В таблице 4 приведен перечень исключительных ситуаций для разрабатываемой системы и описаны реакции системы на их возникновение.

Таблица 3 – Перечень функций, выполняемых системой

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название  подсистемы | Название функции | Информационная среда | | | |
| Входные данные | | Выходные данные | |
| Назначение (наименование) | Тип, ограничения | Назначение (наименование) | Тип, ограничения |
|  |  |  |  |  |  |
| Справочная | Выдать сведения о разработчиках | Сведения о разработчиках системы (ФИО, номер группы) | Текст (МЕМО) | Визуальное отображение информации | – |
| Выдать сведения о системе | Файл справки | Текстовый (\*.HTML) |
| Код ошибки | целое |
| Настройки параметров кроссворда | Подключить словарь понятий | Имя файла | Строка, \*.dict | Список понятий и их определений | Динамический массив строк |
| Код ошибки | Целое |
| Задать количество букв в пересечении | Диапазон количества букв | Целое  1..3 | Количество букв в пересечении | Целое |
| Ввести длину кроссворда | Допустимый диапазон значений | Целое  50..300 | Текущая длина кроссворда | Целое |
| Код ошибки | Целое |

Продолжение таблицы 3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| Файловая | Загрузить файл с кроссвордом | Имя файла | Строка, \*.kros | Кроссворд | Объект «Кроссворд», структура определяется в ходе проектирования |
| Код ошибки | Целое |
| Генерирования | Получить список слов | Маска | Строка | Список слов | Динамический массив строк |
| Список понятий и их определений | Динамический массив строк |
| Авторизации | Ввести логин | Набор допустимых символов | Латинские, русские буквы, цифры, «\_» | Логин | Строка |
| Допустимая длина | Целое  4..10 | Код ошибки | Целое |
| Аутентифировать пользователя | Логин | Строка | Код ошибки | Целое |
| Список данных пользователей | Сущность БД «Пользователи» |

Таблица 4 – Перечень исключительных ситуаций

|  |  |  |
| --- | --- | --- |
| Название  подсистемы | Название исключительной ситуации | Реакция системы |
| 1 Справочная | 1.1 Не возможно открыть файл справки | Выдача сообщения «Файл справки поврежден» |
| 1.2 Не возможно найти файл справки | Выдача сообщения «Отсутствует файл справки» |
| 2 Файловая | 2.1 Попытка открытия файла с несобственным форматом | Выдача сообщения «Файл поврежден или недопустимого формата» |
| … | … | … |

* 1. Разработка прототипа интерфейса пользователя системы

Дать определение интерфейса, отметить основные особенности разработки интерфейса.

Здесь должны быть разработаны прототипы **всех** основных форм приложения с описанием привязанной к ней функциональности, например:

******

***Пример.***

На рисунке ххх приведен прототип экранной формы начальной настройки приложения. Здесь пользователь должен выбрать язык программирования, на котором написан алгоритм, категорию (поиск или сортировка) и нажать кнопку «Далее» для перехода к следующему экрану (форме).

  
Рисунок ххх – Прототип экранной формы начальной настройки приложения

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена навигационная модель разрабатываемого приложения.

* 1. Разработка информационно-логического проекта системы

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

****

Рисунок ХХХ ‒ Навигационная модель приложения

* + 2. Язык UML

Для специфицирования (построения точных, недвусмысленных и полных моделей) системы и ее документирования используется унифицированный язык моделирования UML.

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

* + 1. Диаграмма вариантов использования

Диаграмма вариантов использования представляет собой наиболее общую концептуальную модель сложной системы, которая является исходной для построения всех остальных диаграмм. На ней изображаются отношения между актерами и вариантами использования.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма вариантов использования (пользователя). Здесь должно быть описание диаграммы.

* + 1. Сценарии

Сценарий (scenario) ‑ определенная последовательность действий, которая описывает действия актеров и поведение моделируемой системы в форме обычного текста [27].

В контексте языка UML сценарий используется для дополнительной иллюстрации взаимодействия актеров и вариантов использования.

Рассмотрим несколько сценариев.

Сценарии определяются преподавателем.

Рисунок ХХХ − Диаграмма вариантов использования системы

* + 1. Диаграмма классов

Диаграммы классов – это наиболее часто используемый тип диаграмм, которые создаются при моделировании объектно-ориентированных систем, они показывают набор классов, интерфейсов и коопераций, а также их связи. На практике диаграммы классов применяют для моделирования статического представления системы, они служат основой для целой группы взаимосвязанных диаграмм – диаграмм компонентов и диаграмм размещения [ХХХ].

На рисунке ХХ приведена диаграмма классов системы (этап проектирования). В таблице ХХ приведено описание классов.

Таблица ХХ – Описание классов системы

|  |  |
| --- | --- |
| Название класса | Назначение |
| 1 | 2 |
|  |  |
|  |  |
|  |  |
|  |  |

* + 1. Диаграмма состояний

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма состояний системы. Здесь должно быть описание диаграммы (диаграмм).

* + 1. Диаграмма деятельности

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма деятельности системы. Здесь должно быть описание диаграммы (диаграмм).

* + 1. Диаграмма последовательности

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграммы последовательности системы для варианта использования «???». Диаграммы построены на основании сценариев, приведенных в п.2.4.3.

* 1. Логическая модель данных (при необходимости)

Логическая информационная модель – модель данных, в которой учитывается способ логического хранения данных в памяти ЭВМ. При построении модели базы данных (БД) используются следующие понятия.

Сущность – объект предметной области, который можно отличить от других понятий по некоторым признакам. Сущность состоит из множества своих экземпляров. Каждая сущность обладает свойствами – атрибутами [??].

Атрибут – определенное свойство сущности. Именно набор атрибутов, в общем случае уникальный для каждой сущности, позволяет выделить ее среди других объектов и назвать уникальным именем.

Атрибут или набор атрибутов, используемый для идентификации экземпляра сущности, называется ключом сущности. В случае если для идентификации экземпляра используется один атрибут, ключ называется простым; в противном случае ключ составной. Каждый экземпляр сущности однозначно определяется ключом [??].

Логическая модель БД разрабатываемой системы приведена на рисунке ХХХ.

  
Рисунок 24 – Логическая модель данных

Описание объектов рассматриваемой предметной области, которые хранятся в базе данных, приведено в таблицах 2-???.

Таблица 2 – Сущность «Пользователь»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| Ид пользователя | Целый | Уникальный идентификатор пользователя |
| Имя | Символьный[30] | Имя, используемое при идентификации пользователя и его взаимодействии с системой |
| Пароль | Символьный[10] | Пароль пользователя, преобразованный в закодированную строку |
| Email | Символьный[50] | Электронная почта, указанная пользователем при регистрации |

* 1. Выбор и обоснование алгоритмов обработки данных /Разработка и описание алгоритмов обработки данных

Вводные слова про необходимость разработки алгоритмов.

На рисунке ХХХ приведена схема алгоритма обработки элементов массива. Здесь должно быть краткое описание алгоритма.

  
Рисунок ХХХ – Схема алгоритма обработки элементов массива

На рисунке ХХХХ приведена схема алгоритма вычисления исходного выражения.

1

Начало

a

d

result = 0

push eax

eax = a

imul eax

add eax,1

eax = d

ebx = 4

Рисунок ХХХ – Схема алгоритма вычисления исходного выражения (начало)

Рисунок ХХХХ– Схема алгоритма вычисления исходного выражения (окончание)

sub eax, ebx

ebx = eax

eax = 24

pop ebx

idiv ebx

result

Конец

result = eax

idiv ebx

1

* 1. Выбор и обоснование комплекса программных средств

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

* + 1. Выбор языка программирования

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст. Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

* + 1. Выбор среды программирования

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

* + 1. Выбор операционной системы

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

* + 1. Выбор системы управления базами данных (при необходимости)

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

1. Реализация системы
   1. Разработка и описание интерфейса пользователя

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунках приведены примеры того, как нужно оформить сведения о разработчиках.

  
Рисунок ХХХ – Сведения о разработчиках

  
Рисунок ХХХ – Сведения о разработчиках

* 1. Диаграммы реализации

Диаграммы реализации предназначены для отображения состава компилируемых и выполняемых модулей системы, а также связей между ними. Диаграммы реализации разделяются на два конкретных вида: диаграммы компонентов (component diagrams) и диаграммы развертывания (deployment diagrams) [ХХХ].

* + 1. Диаграмма компонентов

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма компонентов, их описание приведено в таблице ХХХ.

Рисунок ХХХ – Диаграмма компонентов системы

Таблица ХХХ – Описание компонентов системы

|  |  |  |
| --- | --- | --- |
| Название компонента | Назначение компонента | Подсистема |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* + 1. Диаграмма развертывания

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма развертывания системы. Здесь должно быть описание тех компонентов, которые развернуты на узлах ЭВМ.

Рисунок ХХХ – Диаграмма развертывания системы

* + 1. Диаграмма классов

В соответствии со спецификацией, приведенной в п. 2.5.6, и с учетом выбранного языка программирования (см. п. 2.8.1) разработана диаграмма классов системы (этап реализации), приведенная на рисунке ХХХ.

Рисунок ХХХ – Описание классов системы (этап реализации)

* 1. Физическая модель данных (при необходимости)

Физическое проектирование является последним этапом проектирования базы данных, при выполнении которого принимается решение о способах реализации разрабатываемой базы данных. Во время логического проектирования была определена логическая структура базы данных (которая описывает отношения и ограничения в рассматриваемой прикладной области).

Физическая модель базы данных содержит все детали, необходимые конкретной СУБД для создания базы: наименования таблиц и столбцов, типы данных, определения первичных и внешних ключей [??].

На рисунке ХХХ представлена физическая модель данных системы.

Рисунок ХХХ – Физическая модель данных системы

В таблицах ??-?? приведено описание сущностей БД. Первичные ключи выделены жирным шрифтом, а внешние – курсивом.

Таблица ХХХ – Сущность « User »

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Имя атрибута | Тип | Размер (байт) |
| **user Id** | **uniqueidentifier** | **int** | **4** |
| Name | Имя пользователя | varchar(30) | 30 |
| Password | Пароль | varchar(10) | 10 |
| e-mail | Адрес электронной почты | varchar(50) | 50 |
| Размер записи | | | 94 |

* 1. Выбор и обоснование комплекса технических средств
     1. Расчет объема занимаемой памяти

Расчет объема внешней памяти

Для расчета необходимого объема свободной внешней памяти, необходимой для функционирования системы, воспользуемся следующей формулой:

VЖД = VОС + VПР + VСПО + VБД + Vсправки,

где VОС – объем памяти, занимаемый операционной системой (операционная система Windows 7 Professional 64 бит с пакетом обновлений SP1,   
VОС = 20 Гб);

VПР – объем памяти, занимаемый непосредственно файлами приложения (VПР = 2 Мб);

VСПО – объем памяти, занимаемый сопутствующим программным обеспечением (библиотеки cryptopp.dll, simplexlsx.dll, sqlite3.dll, sqlitecpp.dll, Qt Framework 5.11.1, Internet Explorer 9; дадим оценку сверху VСПО в 3 Гб);

VБД – объем памяти, занимаемый базой данных (всеми таблицами) при ее максимальном заполнении. Расчет этой составляющей приведен в таблице ХХХ (VБД = ???? байт = ??? Кб = ??? Мб = ??? Гб).

Vсправки – объем памяти, необходимый для хранения файла справки (Vсправки =0,8 Мб).

Таким образом, суммарный объем внешней памяти составит:

VЖД = 20 Гб + 2 Мб + 3 Гб + ??? Мб + 1 Мб ~ ??? Гб.

Таблица ХХХ – Расчет объема внешней памяти, необходимой для хранения БД

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица | Размер записи (байт) | Максимум записей | Всего (байт) |
| Пользователь | 94 | 10 | 940 |
| Сотрудник |  | 30 |  |
| Статус сотрудника |  | 10 |  |
| Должность сотрудника |  | 10 |  |
| Место работы |  | 10 |  |
| Кафедра |  | 10 |  |
| ОУ ВО |  | 10 |  |
| Итого | | |  |

Расчет объема ОЗУ

Для расчета необходимого объема ОЗУ воспользуемся следующей формулой:

VОЗУ = VОС + VПР + VБД + Vбраузера,

где VОС – ОЗУ, занимаемое операционной системой (2 Гб);

VПР – ОЗУ, которое займет само приложение (не превысит 80 Мб);

VБД – объем данных из базы, который может быть одновременно загружен в оперативную память (дадим ему оценку сверху в 10 Мб).

Vбраузера – ОЗУ, занимаемое браузером (оценим его сверху значением в 100 Мб).

Суммарные объемы ОЗУ составит:

VОЗУ = 2 Гб + 80 Мб + 10 МБ + 100 Мб ~ 2.2 Гб.

Таким образом, 2.2 Гб оперативной памяти можно счесть минимально необходимым для функционирования системы.

* + 1. Минимальные требования, предъявляемые к системе

Для корректного функционирования системы необходимо:

* тип ЭВМ: x86-64 совместимый;
* объем ОЗУ – не менее 3 Гб;
* объем свободного дискового пространства – не менее ??? Гб;
* клавиатура или иное устройство ввода;
* мышь или иное манипулирующее устройство;
* процессор – Intel Pentium не менее 1,5 ГГц;
* дисплей с разрешением не менее 1024 × 768 пикселей;
* операционная система Windows 7 и выше;
* браузер Internet Explorer 9 и выше;
* Qt framework 5.11 и выше.

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта была разработана автоматизированная система …, позволяющая ….

В первом разделе приведены основные понятия предметной области, описаны характеристики систем-аналогов, приведен их сравнительный анализ. На основе проведенного анализа выполнена объектная декомпозиция, отраженная в диаграмме объектов, и сформулирована постановка задачи.

Во втором разделе …

В третьем разделе …

Разработанная система может использоваться …

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

**Книги**

***Целиком***

Буч Г., Рамбо Д., Якобсон А. Язык UML. Руководство пользователя. Изд. 2-е. М.: ДМК Пресс, 2006. 546 с.

*Если нужно указать номера конкретных страниц*

Буч Г., Рамбо Д., Якобсон А. Язык UML. Руководство пользователя. Изд. 2-е. М.: ДМК Пресс, 2006. С. 21.

*Если повторная ссылка на тот же документ*

Буч Г., Рамбо Д., Якобсон А. Язык UML … С. 31.

*Если больше 3 авторов*

Нестационарная аэродинамика баллистического полета/ Липницкий Ю.М. и [др.]. М.: Физматлит, 2003. 176 с.

**Журналы**

Зеленко Л.С., Шумская Е.А. Комплекс программ для работы с учебным контентом в дистанционных обучающих системах// Известия СНЦ РАН. 2015. №2 (5). Т. 17. С. 992-1003.

**Руководящие материалы и ГОСТы**

РД 34.20.571. Методические указания по расчету показателей готовности к работе электростанции и энергосистем. Введ. 1976-10-22. М., 1976. 25 с.

ГОСТ Р 7.0.4-2006. Издания. Выходные сведения. Общие требования и правила оформления. М., 2006. II. 43 с. (Система стандартов по информ., библ. и изд. делу).

**Методические указания или учебные пособия**

Зеленко Л.С. Методические указания к лабораторному практикуму по дисциплине «Программная инженерия». Самара: СГАУ, 2012. 67 с.

**Электронные ресурсы**

Слепой метод печати [Электронный ресурс] // Википедия: [сайт]. URL: [https://ru.wikipedia.org/wiki/slepoi\_metod\_pechyati](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%B5%D0%BF%D0%BE%D0%B9_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4_%D0%BF%D0%B5%D1%87%D0%B0%D1%82%D0%B8%20)(дата обращения: 23.09.2024)

Российская гидроэнергетика [Электронный ресурс] // Русгидро: [сайт]. URL: http://www.rushydro.ru/industry/russianhydropower/ (дата обращения: 20.12.2024).

Гидроэлектростанция (гидроэлектрическая станция, ГЭС) // Энциклопедический словарь юного техника М.: Издательство «Педагогика», 1987 [Электронный ресурс] // Библиотекарь.Ру: электрон. библ. 2006-2024. URL: http://www.bibliotekar.ru/enc-Tehnika/58.htm (дата обращения: 20.12.2024).

Субботин А.С. Основы гидротехники [Электронный ресурс]. URL: http://www.cawater-info.net/bk/dam-safety/files/subbotin.pdf (дата обращения: 03.02.2024).

Филиальная структура компании [Электронный ресурс] // Системный оператор Единой энергетической системы: [сайт]. [2009-2017]. URL: http://so-ups.ru/index.php?id=about (дата обращения: 20.12.2024).

Автоматизированные системы управления технологическими процессами гидроэлектростанции [Электронный ресурс] // Микроника. Инжиниринговый центр: [сайт]. [1999-2016]. URL: http://mikronika-energo.ru/products/asutp/ges-asu-tp/ (дата обращения: 24.12.2024).

Автоматизированная система управления производственными процессами [Электронный ресурс] // MEScontrol: [сайт]. [2003-2017]. URL: http://mescontrol.ru/articles/systems (дата обращения: 02.04.2024).

Пушников А.Ю. Введение в системы управления базами данных: учеб. пособие [Электронный ресурс] // CITForum: электрон. библиотека. 1997-2017. URL: https://citforum.ru/database/dblearn/ dblearn06.shtml (дата обращения: 20.12.2024).

Пользовательский интерфейс [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2017. URL: https://ru.wikipedia.org/ wiki/Пользовательский\_интерфейс (дата обращения: 17.03.2024).

***Если необходимо указать системные требования для доступа к документу (наличие специального ПО), то***

Белова С.В. Язык UML. Диаграмма вариантов использования. Систем. требования: PowerPoint. URL: nkse.ru/component/k2/item/  
download/7\_754f5a247edc6ec6be78218f187338a5.html (дата обращения: 17.10.2024).

**Сборники научных трудов или трудов конференций**

Философия культуры и философия науки: проблемы и гипотезы: межвуз. сб. науч. тр./ Саратов. гос. ун-т; [под ред. С.Ф. Мартыновича]. Саратов: изд-во Сарат. ун-та, 1999. 199 с.

Акимова А.Е., Трешников А.А., Зеленко Л.С. Информационная среда ГЭС. Подсистема расчета показателей эффективности работы оборудования // Перспективные информационные технологии (ПИТ-2017): сб. науч. тр. межд. научно-техн. конф.; [под ред. С.А. Прохорова]. Самара: Изд-во СНЦ РАН, 2017. С. 41-44.

***Если электронное издание***

Акимова А.Е., Трешников А.А., Зеленко Л.С. Подсистема расчета показателей эффективности работы оборудования // Математика. Компьютер. Образование: труды XXIV межд. конф., 23-28 января 2017 г., г. Пущино. URL: http://www.mce.su/rus/presentations/ p283063/ (дата обращения: 02.03.2017).

ПРИЛОЖЕНИЕ А  
Руководство пользователя

А.1 Назначение системы

Приводится краткое описание возможностей системы.

А.2 Условия работы системы

***Пример.***

Для корректной работы системы необходимо наличие соответствующих программных и аппаратных средств.

1. Требования к техническому обеспечению:

* ЭВМ типа IBM PC;
* процессор типа x86 или x64 тактовой частоты 1400 МГц и выше;
* …

1. Требования к программному обеспечению:

* Windows 7 Professional 64 бит с пакетом обновлений SP1 и выше;
* установленная платформа .Net версии 4.0 и выше;
* установленная СУБД ….

А.3 Установка системы

***Пример.***

Система поставляется в виде zip-архива. Данный файл необходимо распаковать в любую директорию на жестком диске. Запускаемым файлом системы является файл ххх.exe.[[2]](#footnote-2)

А.4 Работа с системой

А.4.1 Работа с системой в режиме администратора (если необходимо)

Вход в систему (авторизация)

…

А.4.2 Работа с системой в режиме пользователя

Вход в систему (авторизация)

Вход в систему (регистрация)

Настройка параметров кроссворда

ПРИЛОЖЕНИЕ Б   
Листинг модулей программы

7-10 страниц исходного кода шрифт Times New Roman 10 пт 1 интервал

1. *Количество страниц, рисунков, таблиц указывается с учетом приложений* [↑](#footnote-ref-1)
2. Если необходимы дополнительные ресурсы для обеспечения работоспособности системы, то все для них также должны быть перечислены условия установки. *Если установка нестандартная, то она должна быть подробно описана (в объеме, достаточном для понимания пользователя).* [↑](#footnote-ref-2)